

## 5. INFORMIX

### 5.1 INFORMIX Structured Query Language (SQL)

INFORMIX-SQL is a Relational Database Management System designed to, substantially, reduce the amount of time required to organize, store, and retrieve information. INFORMIX is a computer-based, record-keeping system consisting of useful programs or modules that perform data management tasks. INFORMIX-SQL can summarize, group, and format information in ways that would not, otherwise, be possible.

#### 5.1.1 Database Organization

The INFORMIX database is a collection of information or data that is organized into "tables" and "index" files. A data "table" is an organized collection of defined variables and associated information or data contained in these variables. An index "file" is a predefined list of variables called "keys" that keep track of the rows in a table based on values stored in one or more columns. Index "files" are used to find information or data more quickly. For every table in a database, there will always be a corresponding index. The data "tables" and "index" files are contained in a directory structure. In the database directory structure, there are special files called "system" tables. "System" tables are INFORMIX software tables used by INFORMIX to manage/maintain the database, and contain information on the data "tables" and "index" files. It is important to note that these tables are only to be used by INFORMIX and should not be modified in any way. The result of modification to these tables may cause the deletion of the entire database!

#### 5.1.2 INFORMIX Software

The INFORMIX software is the interface between the operating system and the application database and programs. The INFORMIX software is organized into libraries, which are collections of directories containing files used by INFORMIX, to maintain the database structure; and allow the application programs to access the database to perform some type of processing. In order for other individuals to access the database and execute application programs, "connect" permissions must exist for each authorized database user. Another aspect of INFORMIX is the transaction log. The transaction log contains the "before" and "after" images of modifications to the database. If a modification is unsuccessful (i. e., program/system failure), INFORMIX will restore (put back) the "before" image of the database before the modification was attempted. "SACEGO" is an INFORMIX command that runs a compiled ACE report to generate a hard copy table listing. These table listings are used to verify data output that is the result of program or conversion execution. The SA can use this output as a reference for finding information that may help resolve application problems. INFORMIX also utilizes several environmental variables in each authorized user ".profile" file.

### 5.1.3 Using INFORMIX-SQL

INFORMIX-SQL is utilized by the AFMIS SA to maintain the AFMIS database, monitor the application system, observe user activities, and identify and correct application problems. SQL are stand-alone statements used by the INFORMIX software to perform routine maintenance, correct problems, and monitor application software performance. The format for identifying or creating an SQL is "file1.sql", where the ".sql" extension on the file name, identifies the file as a special purpose INFORMIX file.

- **SELECT Syntax.** The SELECT statement is the most used SQL statement in INFORMIX. This is a "query" of the contents of a table based on one or more conditions. The only requirements in a SELECT statement are the keywords "SELECT" and "FROM." The following syntax represents an expanded SELECT statement with several optional keywords and a detailed explanation of each part of the SQL statement:

```
SELECT [all or specific] FROM [table name]
WHERE [variable] [option(s)] [occurrence or variable]
ORDER BY [occurrence or variable]
```

- **SELECT.** SELECT retrieves all records associated with the chosen specific criteria.
  - "all or specific". "All" may be represented with an "\*" and "specific" can be any combination of one or more field names contained on the table.
  - FROM. A command that indicates/points to the table where the information or data is extracted.
  - "table name". Specific name of the table from where the information or data is extracted.
  - WHERE. An optional clause that fine tunes the SELECT statement to retrieve only the particular occurrence(s) requested.
  - "variable". A value that causes the SELECT statement to choose information or data for the following option(s).
  - "option(s)". An argument that indicates what the variable, being selected, should or should not look like.
  - "occurrence or variable". Specific data value used to narrow select statement criteria.
- **ORDER BY.** An optional clause used to organize selected data for easier review.

- "occurrence or variable". Data value that is used to organize selected data.
- WHERE Syntax.
  - The WHERE clause is an optional clause used to narrow a SELECT statement so only the specific required information is retrieved from the database. There are various ways of building a WHERE clause, as the following examples illustrate.
  - LIKE. Used to check a field on a table for a value that is similar to the specified literal value criteria. "Wild cards" are used to find the information or data.

(i. e.) where [field name] like "value%"

- MATCHES. Used to check a field on a table for a value that equals the specified literal value criteria. "Wild cards" are used to find the information or data.

(i. e.) where [field name] matches "value\*"
   
               where [field name] matches "XYZ"

- NULL. Used in conjunction with finding fields that do not hold values. This is not a zero value but considered as a "non-value" to the system.

(i. e.) where [field name] IS NULL

- NOT NULL. Used in conjunction with finding fields that contain unknown values.

(i. e.) where [field name] IS NOT NULL

- = . Used to select information or data that is equal to the specified literal value criteria. This is a comparison between two values.

(i. e.) where [field name] = "value"

- != . Used to select information or data that is not equal to the specified literal value criteria. This is a comparison between two values.

(i. e.) where [field name] != "value"

- TODAY. Used in conjunction with retrieving data from a table based on the current system date.

(i. e.) where [field name] << TODAY  
 where [field name] <= TODAY  
 where [field name] = TODAY  
 where [field name] >= TODAY  
 where [field name] > TODAY

- AND. Used to join two or more WHERE clause option statements. Both statements must select true or no data will be retrieved.

(i. e.) where [field name] = "value" and  
 [field name] IS NOT NULL

- OR. Used to join two or more WHERE clause option statements. The OR is similar to the AND but will select data if either statement is true.

(i. e.) where [field name] <= TODAY or  
 [field name] != "value" or  
 [field name] matches "[XYZ]"

- SELECT COUNT(\*) Syntax.

- The SELECT COUNT(\*) statement is used to retrieve a whole number count of all the records contained on a database table. The SA can use a WHERE clause in conjunction with the SELECT COUNT(\*) statement to narrow the search, if desired.

(i. e.) SELECT COUNT(\*) FROM [table name]

- SQL Manipulation Statements.

- The following three statements are only used by the SA under the direction of the AFMIS Customer Assistance Office (CAO) to manipulate the data in a database table.

- INSERT. Adds record(s) to the table.
- UPDATE. Modifies record(s) or values on the table.  
Must use WHERE clause.
- DELETE. Removes the specified record(s) from the table.  
Must use WHERE clause.

- Comments in SQL Statements. The history of each SQL should contain the following:
  - • {}. Used to enclose comments in SQL statements and placed in the first column of the first line of the file. The comments are placed in between the "{}".

(i. e.) {grant.sql - connects authorized users to the database}

- Name. The descriptive name of the SQL that indicates what the SQL will do.
- Comments. Comments are a detailed explanation of what the SQL will do during execution and any related information that is necessary.

- Running SQL.

SQL statements are "stand-alone" code that can be run in background in order to free the SA's terminal for further work.

- • Executing SQL. SQL may be run straight from the command line or by a "script".  
Output will be sent to the screen or to a named file.

(i. e.) isql afmis filename  
isql afmis filename >out1

- • Executing SQL in Background. SQL run in background will free the terminal for other work, and the output will be placed in named files for informational purposes.

(i. e.) nohup isql afmis filename >out1 2>out2 &

- nohup. No hang up. Even if terminal is logged-off, the SQL will continue to execute.
- > This symbol tells the system to "redirect" (place) the "messages" (output) generated during execution into a file.
- 2> Sends "error out" to a file informing the SA if there were any problems or if the SQL was successful. The SA may name these files in any manner chosen, however, remember that each SQL is run for a specific reason and output files are named, accordingly, if the same SQL is run several times in a row but for different reasons.
- & Tells the system that this SQL will be run in background.

#### 5.1.4 Database Administration SQL

The SA may be required to execute some specialized SQL in order to efficiently maintain the AFMIS database. The following statements will be run under the guidance of CAO:

- UNLOAD.

The UNLOAD statement is used to copy data records from a table to a UNIX flat file. The purpose of this is to ensure data is not corrupted while conducting routine database maintenance.

```
UNLOAD TO "path name or file name"  
SELECT [all or specific]  
FROM [table name]  
WHERE [field name] = "value"
```

- LOAD.

The LOAD statement is used to copy data records into a data table from a UNIX flat file that has been previously UNLOADED from a data table. The BEGIN WORK and COMMIT WORK statements around the LOAD are known as a transaction. A transaction is an INFORMIX safety feature designed to protect the data table from corruption if a system failure occurs during a data manipulation statement.

```
BEGIN WORK;  
LOAD FROM "path name or file name"  
INSERT INTO [table name];  
COMMIT WORK
```

- CHECK TABLE.

The CHECK TABLE statement checks the indexes of a table to determine if they have become corrupted. An error message will be produced if the system detects any irregularities in the index file.

```
CHECK TABLE [table name];  
CHECK TABLE [table name];  
CHECK TABLE [table name];  
CHECK TABLE [table name]
```

- REPAIR TABLE.

THE REPAIR TABLE statement repairs the indexes (in most cases) of a table if they have become corrupted.

```
REPAIR TABLE [table name];  
REPAIR TABLE [table name];  
REPAIR TABLE [table name];  
REPAIR TABLE [table name]
```

- GRANT.

The GRANT statement will provide access privileges to the list of authorized users on the proper database.

```
GRANT CONNECT TO
```

```
user name ,  
user name ,  
user name ,  
user name ,  
user name
```

- REVOKE.

The REVOKE statement will remove access privileges on the database for a list of specified users.

```
REVOKE CONNECT FROM
```

```
user name ,  
user name ,  
user name ,  
user name ,  
user name ,  
user name
```

- DROP TABLE.

The DROP TABLE statement is used to remove a table from the database.

```
DROP TABLE [table name]
```

- CREATE TABLE.

The CREATE TABLE statement is used to create a table or recreate one that has been dropped. The "NOT NULL" is used ONLY if the field does not allow null values.

```
CREATE TABLE [table name]
(
    [field name]      [data type](size)(NOT NULL),
    [field name]      [data type](size),
    [field name]      [data type](size)
)
```

## 5.2 TISA/TISA-W/IFA Database

The TISA/IFA database is composed of interactive tables used in conjunction with the TISA, IFA, and TISA-W subsystem application software. Of the tables, 12 are INFORMIX system tables designed to keep track of tables, columns, indexes, views, synonyms, and permissions on the database. The remaining tables are designed to hold all information (data) required to process the TISA/IFA software applications.

### 5.2.1 TISA/TISA-W/IFA Database Location

The TISA/IFA database is located on the "/informix/tisa" file system in the directory "afmis.dbs". The ".dbs" extension identifies this directory as a database to both the application and INFORMIX software. In the "afmis.dbs" directory, there are two types of files, ".dat" files and ".idx" files. The ".dat" extension designates that file as a table that contains the applications data. The ".idx" extension contains index information for each corresponding data file. There are several INFORMIX system data tables and their corresponding index files identifiable by prefix "sys." They are designed to keep track of data tables, columns, indexes, views, synonyms, and permissions on the database. Figure 5.2-1 lists each of the TISA/IFA database tables by ID and name. Permissions of the files in the "afmis.dbs" directory are "read", "write", and "execute" for the owner ("afmis"), and group ("informix"), and "read" only for any other valid system logins.

<b><u>Database Table</u></b>	<b><u>Table Name</u></b>
BDFA	BDFA COMPUTATION ITEMS
CAH	CUSTOMER ACCOUNT HEADER
CAT	CUSTOMER ACCOUNT TRAILER
CDN	CUSTOMER DOCUMENT NUMBER
CIF	CUSTOMER INFORMATION FILE
CINC	CUSTOMER INFORMATION NOT CURRENT
COF	CUSTOMER ORDER FILE
COM	COMMUNICATIONS FILE
CRF	CANDIDATE REQUISITION FILE
CSHC	COMMON SERVICES HEADCOUNT
CTL	REPORT CONTROL
CXV	CALL ORDER CROSS REFERENCE VRGC NUMBER
DAS	DINING FACILITY ACCOUNT STATUS
DFC	DINING FACILITY CONTRACT
DFCD	DINING FACILITY CLOSED DATES
DFE	DELIVERY FREQUENCY FILE
DFX	DINING FACILITY EXTENSION
DHF	DOCUMENT HISTORY FILE
DHI	DOCUMENT HISTORY INPUT
DPSC	DPSC CHANGES FOR MIF NSN
DSC	DESIGN CAP FILE
DHO	DOCUMENT HISTORY OUTPUT
H3161	3161 HEADER
ILH	ISSUE LIST HEADER
IUE	INSTALLATION UNIQUES EXTERNALS
IUF	INSTALLATION UNIQUES FILE
IUF2	INSTALLATION UNIQUES FILE #2
IUF3	INSTALLATION UNIQUES FILE #3
LPO	LOCAL PURCHASE ORDER
LPP	LOCAL PURCHASE PRICE
MCI	MEAL COST INFORMATION
MIF	MASTER ITEM FILE
MMF	MASTER MENU FILE
MMR	MASTER MENU RECAP
OEF	OBLIGATION ESTIMATE FILE
OHC	OTHER HEADCOUNT
OIS	OTHER ISSUES 2969
PRC	PRICE TABLE

**FIGURE 5.2-1 - TISA Database Tables**

<b><u>Database Table</u></b>	<b><u>Table Name</u></b>
RCF	RECEIPT CONTROL FILE
RCH	RECEIPT CONTROL HEADER
RCL	REPORT CONTROL LOG
REF	MASTER ITEM REFERENCE
RHC	REIMBURSABLE HEADCOUNT 2969
RHF	RECIPE HEADER FILE
RIF	RECIPE INSTRUCTION FILE
RIN	RECIPE INGREDIENT FILE
RISA	REIMBURSABLE ISSUES AND SALES ACCOUNT
RSFA	REIMBURSABLE SPECIAL FOOD ALLOWANCE
SCC	SOURCE CODE CHANGE
SFA	SPECIAL FOOD ALLOWANCE
SYSUSERS	USERS WITH CONNECT PERMISSIONS
SIF	STARFIARS INTERFACE FILE
T3161	3161 TRAILER
TIN	TRANSACTION INPUT
TOT	TRANSACTION OUTPUT
TRF	TRANSACTION REGISTER FILE
VIF	VENDOR INFORMATION FILE
VOF	VENDOR ORDER FILE
VRGC	VOUCHER REGISTER AND GENERAL CONTROL

**FIGURE 5.2-1 - TISA Database Tables (CONT)**

### **5.3 DFO Database**

The DFO database is composed of interactive data tables and indexes used in conjunction with the DFO subsystem application software. Of the data tables and indexes, 10 are INFORMIX system tables and indexes designed to keep track of data tables, columns, indexes, views, synonyms, and permissions on the database. Figure 5.2-2 lists each of the DFO database tables by ID and name. The remaining data tables and indexes are designed to hold all information (data) that is required to process automated mission procedures for each installation.

#### **5.3.1 DFO Database Location**

The DFO database is located on the "/Informix/dfo" file system in the directory, /informix/dfo/afmis/afmisdb.dbs. The ".dbs" extension identifies this directory as a database to the application and INFORMIX software. In the "afmisdb.dbs" directory, there are two types of files, ".dat" files and ".idx" files. The ".dat" extension designates that file as a table and the ".idx" extension contains information on the indexes for each table. Preceding each of these two extensions, there is a unique character table name, while system tables and indexes always begin with a prefix of "sys". Permission of the files in the "afmisdb.dbs" directory are "read", "write", and "execute" for the owner ("afmis"), and group ("informix"), and "read" and "write" only for any other valid system logins.

<b><u>Database Table</u></b>	<b><u>Table Name</u></b>
ACCT_HEAD	ACCOUNT HEADER TABLE
ACCT_TRL	ACCOUNT TRAILER TABLE
BATCHRPT	DFO BATCH REPORT TABLE
BBD	BULLETIN BOARD DATA
BDFA_INFO	BASIC DAILY FOOD ALLOWANCE INFORMATION TABLE
CASH_TURNIN	CASH TURN-IN TABLE
CROSS	DFO IDENTIFICATION TABLE
CURRENT_DATA	CURRENT DATA TABLE
DF0000IN	DF INPUT
DF0000OUT	DF OUTPUT
ERF	EQUIPMENT REPLACEMENT FILE
H3161	3161 TABLE
ISSFREQ	ISSUE FREQUENCY TABLE
KITREQN	KITCHEN REQUISITION TABLE
MEAL_CST	MEAL COST TABLE
MENUHEAD	MENU HEADER TABLE
MENUTRAIL	MENU TRAILER TABLE
MIF	MASTER ITEM FILE
MIFINV	DFO INVENTORY TABLE
MISC_DATA	MISCELLANEOUS DATA TABLE
ML_PROJ	MEAL PROJECTIONS
ML_12_COST	MEAL COST 12 TABLE
MSTMENU	MASTER MENU TABLE
NWD	NON WORKING DAYS
RCPHEAD	RECIPE HEADER TABLE
RCPINGR	RECIPE INGREDIENT TABLE
RCPINST	RECIPE INSTRUCTION TABLE
SHOPLIST	SHOPPING LIST
SLSTAUS	SHOPPING LIST STATUS
SYSUSERS	USERS WITH CONNECT PERMISSIONS
TOT_HC_2969	TOTAL HEADCOUNT 2969

**FIGURE 5.2-2 - DFO Database Tables**